Logic-Based Benders Methods for Planning and Scheduling

John Hooker
Carnegie Mellon University
August 2003
The Problem:
Multiple-machine resource-constrained scheduling

• Schedule jobs on several machines.
  • Each job consumes some resource at a given rate.
  • Resource consumption rate on each machine must not exceed capacity.
  • Special case: jobs do not overlap.

• Apply logic-based Benders decomposition
  • Generalizes Benders to any subproblem (not just LP, NLP).
  • Solve with MILP and CP (constraint programming).
Basic Idea

• **Decompose** problem into

  assignment + resource-constrained scheduling
  assign jobs schedule jobs on
to machines each machine

• Use **logic-based Benders** scheme to link these.
• Solve: master problem with **MILP**
  -- good at resource allocation
  subproblem with **CP**
  -- good at scheduling

• Applications in manufacturing/supply chain planning & scheduling.
Previous Work

1989 (Jeroslow & Wang) – View Horn SAT dual as inference problem.

1995 (JH & Yan) – Apply logic-based Benders to circuit verification.
  • Better than BDDs when circuit contains error.

  • Specialized Benders cuts must be designed for each problem class.
  • Branch-and-check proposed.

2001 (Jain & Grossmann) – Apply logic-based Benders to multiple-machine scheduling using CP/MILP.
  • Several orders of magnitude speedup wrt CPLEX, ILOG Scheduler.
  • But… easy problem for Benders approach (min cost).
2001 (Thorsteinsson) – Apply branch-and-check to CP/MILP.
   • 1-2 orders of magnitude speedup on multiple machine scheduling.

2002 (JH, Ottosson) – Apply logic-based Benders to SAT, IP.

   • Not useful computationally for planning and scheduling.
   • A simpler relaxation works better.

Today – Apply logic-based Benders to resource-constrained planning/scheduling problems.
   • Multiple machines, parallel processing with resource constraint on each machine
   • Min cost and min makespan.
Single-machine resource-constrained scheduling

- $d_j =$ duration of job $j$
- $r_j =$ rate of resource consumption of job $j$
- $L =$ resources available
- $[a_j, b_j] =$ time window for job $j$

Total resource consumption $\leq L$ at all times.
Multiple-machine resource-constrained scheduling

\[ d_{ij} = \text{duration of job } j \text{ on machine } i \]

\[ r_{ij} = \text{resource consumption of job } j \text{ on machine } i \]

\[ L_i = \text{resources available on machine } i \]

Total resource consumption \( \leq L_i \) at all times.
Some Possible Objectives

Minimize cost = \( \sum_{ij} c_{ij} x_{ij} \)
\( = 1 \) if job \( j \) assigned to machine \( i \)
Cost of assigning job \( j \) to machine \( i \)

Minimize makespan = \( \max_{ij} \{ t_j + d_{ij} \} \)
Start time of job \( j \)

Minimize tardiness = \( \sum_{ij} \max \{ 0, t_j + d_{ij} - b_j \} \)
Deadline for job \( j \)
Minimize Cost: Discrete Time MILP Model

\[
\begin{align*}
\text{min} & \quad \sum_{ijt} c_{ij} x_{ijt} \\
\text{subject to} & \quad \sum_{ijt} x_{ijt} = 1, \quad \text{all } j \\
& \quad \sum_{ij} \sum_{t} r_{ij} x_{ijt} \leq L_i, \quad \text{all } i, t \\
& \quad x_{ijt} = 0, \quad \text{all } j, t \text{ with } b_j - d_{ij} < t \leq a_j \\
& \quad x_{ijt} = 0, \quad \text{all } j, t \text{ with } t > N - d_{ij} + 1 \\
& \quad x_{ijt} \in \{0,1\}
\end{align*}
\]

- = 1 if job \( j \) starts at time point \( t \) on machine \( i \) \((t = 1, \ldots, N)\)
- Job \( j \) starts at one time on one machine
- Jobs underway at time \( t \) consume \( \leq L_i \) in resources
- Jobs observe time windows
Minimize Cost: Discrete Event MILP Model

\[ \text{min} \sum_{ijk} c_{ijk} x_{ijk} \]

subject to

\[ \sum_{ik} x_{ijk} = 1, \quad \sum_{ik} y_{ijk} = 1, \quad \text{all } j \]

\[ \sum_{ij} x_{ijk} + y_{ijk} = 1, \quad \text{all } k \]

\[ \sum_{k} x_{ijk} = \sum_{k} y_{ijk}, \quad \text{all } i, j \]

\[ t_{i,k-1} \leq t_{ik} \]

Events in chronological order

\[ = 1 \text{ if event } k \text{ is start of job } j \text{ on machine } i \quad (k = 1, \ldots, 2N) \]

\[ = 1 \text{ if event } k \text{ is end of job} \]

Each job is assigned to one machine and starts once and ends once

Start time of event \( k \) (disaggregated by machine)
Release date and deadline

\[ a_j x_{ijk} \leq t_{ik}, \quad f_{ij} \leq b_j, \quad \text{all } i, j, k \]

Finish time of job \( j \) (disaggregated by machine)

\[ t_{ik} + d_{ij} x_{ik} - M (1 - x_{ijk}) \leq f_{ij} \leq t_{ik} + d_{ik} x_{ijk} + M (1 - x_{ijk}), \quad \text{all } i, j, k \]

Definition of finish time

\[ t_{ik} - M (1 - y_{ijk}) \leq f_{ij} \leq t_{ik} + M (1 - y_{ijk}), \quad \text{all } i, j, k \]

Resource limit

\[ R_{ik} \leq L_i, \quad \text{all } i, k \]

\[ R_{i1} = R_{i1}^s, \quad R_{ik}^s = \sum_j r_{ij} x_{ijk}, \quad R_{ik}^f = \sum_j r_{ij} y_{ijk}, \quad \text{all } i, k \]

Calculation of resource consumption on machine \( i \) at time of each event

\[ R_{ik}^s + R_{i,k-1} - R_{ik}^f = R_{ik}, \quad \text{all } i, k \]

\[ x_{ijk}, y_{ijk} \in \{0,1\} \]
CP: The Cumulative Constraint

\[
\text{cumulative}
\begin{pmatrix}
(t_1, \ldots, t_n) \\
(d_1, \ldots, d_n) \\
(r_1, \ldots, r_n) \\
L
\end{pmatrix}
\text{ is equivalent to }
\sum_{j} r_j \leq L , \quad \text{all } t
\]
\[t_j \leq t < t_j + d_{ij}\]

Schedules jobs at times \( t_1, \ldots, t_n \) so as to observe resource constraint.

Edge-finding algorithms, etc., reduce domains of \( t_j \).
Minimize Cost: CP Model

\[
\min \sum_j c_{y_j,j}
\]

subject to

\[
\text{cumulative}\left(\begin{array}{l}
(t_j \mid y_j = i) \\
(d_{ij} \mid y_j = i) \\
(r_{ij} \mid y_j = i) \\
L_i
\end{array}\right), \quad \text{all } i
\]

\[
a_j \leq t_j \leq b_j - d_{y_j,j}, \quad \text{all } j
\]

\(y_j = \text{machine assigned to job } j\)

\(\text{start times of jobs assigned to machine } i\)

\(\text{Observe time windows}\)

\(\text{Observe resource limit on each machine}\)
This is how it looks in OPL Studio…

[Declarations]
DiscreteResource machine [i in Machines] (Limit [i]);
AlternativeResources mset(machine);
Activity sched [j in Jobs];

minimize
  sum(j in Jobs) cost[j]
subject to {
  forall(j in Jobs) {
    sched[j] requires(jobm[i,j].resource) mset;
    forall(i in Machines)
      activityHasSelectedResource(sched[j],mset,machine[j])
      <=> sched[j].duration = jobm[i,j].duration &
      cost[j] = jobm[i,j].cost;
    sched[j].start >= job[j].release;
    sched[j].end <= job[j].deadline;
  };
};

search {
  assignAlternatives;
  setTimes;
};

enforces cumulative
assigns jobs to machines
defines resource requirements
determines cost and
durations on the assigned
machine
time windows
invokes specialized search procedure
(needed for good performance)
Inference Duality

Primal

\[
\begin{align*}
\min & \quad f(x) \\
\text{subject to} & \quad x \in S
\end{align*}
\]

Dual

\[
\begin{align*}
\max & \quad z \\
\text{subject to} & \quad x \in S \Rightarrow f(x) \geq z \\
\text{domain} & \quad x \in D
\end{align*}
\]

Solution is a value for \( x \).
Solution is a \textit{proof} that deduces \( f(x) \geq z \) from \( x \in S \).

Duality theorem based on completeness proof for inference method.
Example of Inference Duality: Linear Programming

**Primal**

\[
\begin{align*}
\text{min} & \quad cx \\
\text{subject to} & \quad Ax \geq b \\
x & \geq 0
\end{align*}
\]

**Dual**

\[
\begin{align*}
\text{max} & \quad z \\
\text{subject to} & \quad Ax \geq b \\
& \implies cx \geq z
\end{align*}
\]

Solution is a *proof* that deduces \( cx \geq z \) from \( Ax \geq b \), encoded as a vector of multipliers \( u \).

**Farkas Lemma**

(Completeness theorem for nonnegative linear combination as an inference method)

\[
Ax \geq b \implies cx \geq z \quad \text{iff} \quad uA \leq c, \; ub \geq z \quad \text{for some } u \geq 0
\]

(provided primal or dual is feasible).

Farkas Lemma provides basis for classical LP duality theorem.

Since \( u \) has polynomial size, LP belongs to NP and co-NP.
Logic-Based Benders Decomposition

\[
\begin{align*}
\text{min} & \quad f(x, y) \\
\text{subject to} & \quad (x, y) \in S \\
& \quad x \in D_x, y \in D_y
\end{align*}
\]

**Master Problem**

\[
\begin{align*}
\min_{x,z} & \quad z \\
\text{subject to} & \quad z \geq B_{\bar{x}}^k(x), \text{ all } k \\
& \quad x \in D_x, y \in D_y
\end{align*}
\]

**Subproblem**

\[
\begin{align*}
\min_y & \quad f(\bar{x}, y) \\
\text{subject to} & \quad (\bar{x}, y) \in S \\
& \quad y \in D_y
\end{align*}
\]

Solution of subproblem’s inference dual is a proof of bound

\[ z \geq B_{\bar{x}}(\bar{x}) \] that is valid when \( x = \bar{x} \)

Benders cut is based on bound obtained from the same proof schema for other values of \( x \).
Example: Classical Benders

\[
\begin{align*}
\min & \quad f(x) + cy \\
\text{subject to} & \quad g(x) + Ay \geq b \\
& \quad x \in D_x, y \geq 0
\end{align*}
\]

**Master Problem**

\[
\begin{align*}
\min_{x,z} & \quad z \\
\text{s.t.} & \quad z \geq f(x) + u^k(b - g(x)), \text{ all } k \\
& \quad x \in D_x
\end{align*}
\]

**Subproblem**

\[
\begin{align*}
\min_y & \quad f(\bar{x}) + cy \\
\text{s.t.} & \quad Ay \geq b - g(\bar{x}) \\
& \quad y \geq 0
\end{align*}
\]

Solution of subproblem’s inference dual is a proof (encoded by \(u\)) of bound
\[
z \geq f(\bar{x}) + u(b - g(\bar{x}))
\]

Benders cut is based on bound obtained from \(u\), which remains dual feasible (i.e., proof remains valid) for other values of \(x\).
Minimize Cost: Logic-Based Benders

Master Problem: Assign jobs to machines

\[
\begin{align*}
\text{min} & \quad \sum_{ij} c_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{i} x_{ij} = 1, \quad \text{all } j \\
& \quad \sum_{i} d_{ij} r_{ij} x_{ij} \leq L_i \max_{j} \{b_j\}, \quad \text{all } i
\end{align*}
\]

Benders cuts

Relaxation of subproblem: “Area” \( d_{ij} r_{ij} \) of jobs assigned to a machine fit in the space available before latest deadline.
**Subproblem: Schedule jobs assigned to each machine**

Solve by constraint programming

Let $J_{ik} = \text{set of jobs assigned to machine } i \text{ in iteration } k$.

If subproblem $i$ is infeasible, solution of subproblem dual is a proof that not all jobs in $J_{ik}$ can be assigned to machine $i$. This provides the basis for a (trivial) Benders cut.
Master Problem with Benders Cuts
Solve by MILP

\[
\begin{align*}
\text{min} & \quad \sum_{ij} c_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{j} x_{ij} = 1, \quad \text{all } j \\
& \quad \sum_{j} d_{ij} r_{ij} x_{ij} \leq L_i \max_{j} \{b_j\}, \quad \text{all } i \\
& \quad \sum_{j \in J_{ik}} (1 - x_{ij}) \geq 1, \quad \text{all } i, k \\
x_{ij} & \in \{0,1\}
\end{align*}
\]
**Important observation:** Putting a relaxation of subproblem in the master problem is essential for success.

<table>
<thead>
<tr>
<th>Master Problem</th>
<th>Subproblem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective Function</td>
<td>Computed in master problem</td>
</tr>
<tr>
<td>Subproblem</td>
<td>Optimization problem, more interesting cuts (harder for CP)</td>
</tr>
<tr>
<td>Feasibility Problem, Min makespan</td>
<td>More interesting, nice duality with cuts</td>
</tr>
</tbody>
</table>

**Min cost problem is particularly easy for logic-based decomposition:**

master problem is essential for success.

Putting a relaxation of subproblem in the
Minimize Makespan: Logic-Based Benders

**Master Problem: Assign jobs to machines**

\[
\begin{align*}
\text{min} & \quad T \\
\text{subject to} & \quad \sum_{i} x_{ij} = 1, \quad \text{all } j \\
& \quad \sum_{i} d_{ij} r_{ij} x_{ij} \\
& \quad T \geq \frac{\sum_{j} L_{i}}{L_{i}}, \quad \text{all } i \\
\end{align*}
\]

Benders cuts

Relaxation of subproblem: “Area” of jobs provides lower bound on makespan.
Subproblem: Schedule jobs assigned to each machine
Solve by constraint programming

$$\min \quad T$$

subject to

\[
\begin{align*}
T & \geq t_j + d_{ij}, \quad \text{all } j \\
\text{cumulative} & \begin{cases}
(t_j \mid \bar{x}_{ij} = 1) \\
(d_{ij} \mid \bar{x}_{ij} = 1) \\
(r_{ij} \mid \bar{x}_{ij} = 1) \\
L_i
\end{cases}, \quad \text{all } i \\
\end{align*}
\]

\[
\begin{align*}
a_j & \leq t_j \leq b_j, \quad \text{all } j
\end{align*}
\]

Let $J_{ik}$ = set of jobs assigned to machine $i$ in iteration $k$.

We get a Benders cut even when subproblem is feasible.
**Duality of Linear Relaxation and Linear Benders Cuts**

**Relaxation:**
- Lower bound on makespan

**Benders cut:**
- Lower bounds on makespan as jobs are removed from machine

Minimum makespan for machine $i$ in subproblem
Master Problem: Assign jobs to machines
Solve by MILP

\[
\begin{align*}
\text{min} & \quad T \\
\text{subject to} & \quad \sum_{i} x_{ij} = 1, \quad \text{all } j \\
& \quad \sum_{j} d_{ij} r_{ij} x_{ij} \\
& \quad T \geq \frac{\sum_{j} d_{ij} r_{ij} x_{ij}}{L_i}, \quad \text{all } i \\
& \quad T \geq T_k - \sum_{j \in J_{ik}} (1 - x_{ij}) d_{ij} \\
x_{ij} \in \{0, 1\}
\end{align*}
\]
## Experimental Design

<table>
<thead>
<tr>
<th></th>
<th>2 Machines</th>
<th>3 Machines</th>
<th>4 Machines</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CP (ILOG Scheduler)</strong></td>
<td>6, 8, 10, ... jobs*</td>
<td>6, 8, 10, ... jobs</td>
<td>6, 8, 10, ... jobs</td>
</tr>
<tr>
<td><strong>MILP (Discrete time model)</strong></td>
<td>6, 8, 10, ... jobs</td>
<td>6, 8, 10, ... jobs</td>
<td>6, 8, 10, ... jobs</td>
</tr>
<tr>
<td><strong>Hybrid</strong></td>
<td>6, 8, 10, ... jobs</td>
<td>6, 8, 10, ... jobs</td>
<td>6, 8, 10, ... jobs</td>
</tr>
</tbody>
</table>

*1 problem instance each

** Discrete event model solved none of the problems
"Min Cost, 2 Machines"

Logarithmic scale

- CP
- MILP
- Hybrid

○ = computation terminated
Min Cost, 2 Machines
Linear scale

Sec

CP
MILP
Hybrid

Jobs
Min Cost, 3 Machines

Logarithmic scale

Sec

Jobs

CP
MILP
Hybrid
Min Makespan, 2 Machines

Logarithmic scale

Jobs

Sec

CP

Hybrid

S e c

5 7 9 11 13 15 17
Min Makespan, 3 Machines
Logarithmic scale

CP
Hybrid

Jobs
Sec
Min Makespan, 4 Machines

Logarithmic scale

Sec

Jobs

CP
Hybrid
Min Tardiness

?